

*Nun, da sich der Vorhang der Macht von der Bühne hebt,
kann das Spiel beginnen, das uns vom Drama einer Kultur berichtet.*

*DSGVO und BDSG
WWW und DDG – ojemine*

*DMA, DSA und DGA
NIS und CRA – tatütata*

*EU, BSI und BSIG – MfG!
Mit freundlichen Grüßen*

Frei nach „Die Fantastischen Vier – MfG“



Quelle: www.ask-mediendesign.de

SQL Injection

SQL Server im BSI IT-Grundschutz



Zur Person

■ Bernd Jungbluth

- Freiberuflicher Berater, Entwickler und Coach
 - Access und SQL Server
 - SQL Server Administration und Datenbankentwicklung
 - Datawarehouse-Systeme, Integration Services, Reporting Services
 - Azure SQL Database
- Informationssicherheit, Datensicherheit und Datenschutz
 - Zertifizierter Datenschutzbeauftragter
 - Zertifizierter IT-Grundschutz-Praktiker (BSI)

■ Agenda

- SQL Injection
- Maßnahmen zur Vermeidung von SQL Injection

SQL Injection

- Demo

- SQL Injection

SQL Injection

■ SQL Injection

- Einschleusen von SQL-Anweisungen in die aktuell auszuführende SQL-Anweisung
- Erster veröffentlichter Angriff im Jahr 2000
- 2025 Platz 2 der 25 Most Dangerous Software Weaknesses der [CWE](#)
- Aktuell gemeldete Schwachstellen auf [CVE](#) – Suchbegriff CWE-89
- Benannte Gefährdungslage im BSI IT-Grundschutz APP.4.3 Relationale Datenbanken

■ Angriffsziele

- Datenextraktion
- Manipulation bzw. Löschen von Daten
- Erweitern der Zugriffsrechte und Systemrechte
- Kontrollfluss-Manipulation zum Umgehen von Logik
- Verletzen mehrere Schutzziele der Informationssicherheit

SQL Injection / Informationssicherheit

■ Schutzziele der Informationssicherheit

- Vertraulichkeit ~ Zugang zu Informationen nur für vorgesehene Personen
- Integrität ~ Unversehrtheit von Informationen
- Verfügbarkeit ~ Zugriff auf Informationen im erforderlichen Zeitraum

■ Informationssicherheit

- Umfassender Schutz aller Informationen eines Unternehmens
 - Digitale wie analoge, visuelle und gesprochene Informationen
- IT-Sicherheit – technischer Schutz der IT-Systeme
- Datensicherheit – technischer Schutz aller Daten eines Unternehmens
 - Primäres Ziel der SQL Injection
 - Möglich über verschiedene Angriffswege (Kanäle) wie In-Band, Blind und Out-Of-Band

SQL Injection / Sicherheitsmaßnahmen

■ Einheitliches Angriffsmuster

- Erweitern der aktuell auszuführenden SQL-Anweisung
 - Mit einer vom Angreifenden kontrollierten SQL-Anweisung
- Verändert die Struktur und die Aufgabe der ursprünglichen SQL-Anweisung
- Möglich durch dynamisches SQL sowie zu hohen Berechtigungen

■ Sicherheitsmaßnahmen

- Ausgabe einfacher Fehlermeldungen ohne technische Details
 - Speichern technischer Details in Tabelle oder externer Datei
- Prüfen der Eingaben nach Länge und Inhalt
- Verwenden parametrisierter SQL-Anweisungen
 - Vermeiden dynamischer SQL-Anweisungen
- Berechtigungskonzept

SQL Injection / Sicherheitsmaßnahmen / **Error-Based Injection**

- Demo

- Sicherheitsmaßnahme Error-Based Injection

■ Eingabeprüfung

- Allgemeine Empfehlungen vom [BSI](#), [STIG](#), [CIS](#) und [OWASP](#)
- Prüfen der Länge einer fachlich legitimen Eingabe
 - Beispiel: PLZ mit 7 Stellen bei einer Eingabe von `55469 + ' OR 1 = 1`
- Prüfen auf verbotene Begriffe wie DROP, UNION, UPDATE etc.
 - Beispiel: Eingabe `Meier%' UPDATE Mitarbeiter SET AktuellesGehalt = ...`

■ Clientseitige bzw. serverseitige Prüfung

- Clientseitige Prüfung der Eingabe vor der Übergabe der SQL-Anweisung an die Datenbank
- Serverseitige Prüfung der Eingabe vor dem Ausführen der eigentlichen SQL-Anweisung
- Abbruch mit Fehler bei unkorrekter Länge
- Abbruch mit Fehler bei Eingabe mit enthaltenen verbotenen Begriffen

SQL Injection / Sicherheitsmaßnahmen / Eingabeprüfung

- Demo

- Sicherheitsmaßnahme Clientseitige Eingabeprüfung

SQL Injection / Sicherheitsmaßnahmen / Eingabeprüfung

■ Eingabeprüfung der Länge

- Ergänzende Maßnahme
- Nur wirksam bei erwartbaren kurzen Zeichenfolgen
 - Beispiel: PLZ mit 7 Stellen bei einer Eingabe von `55469 + ' OR 1 = 1`
- Wenig Wirkung bei erwartbaren Zeichenfolgen mit mittlerer bis hoher Länge
 - Beispiel: Spalte Firma mit Datentyp NVARCHAR(255)

■ Eingabeprüfung nach verbotenen Inhalten

- Ergänzende Maßnahme
- Schwierige Umsetzung einer Blacklist durch legitimes Vorkommen verbotener Inhalte
 - Beispiel Firma „Socken **Union** GmbH“
 - Beispiel Produkt „Hy**drop**flanze“
 - Beispiel Kennwort „LegendärerMic**drop**DerNeuro**login**Bei#WaltersReunionParty2026!“

SQL Injection / Sicherheitsmaßnahmen / Parametrisierte Abfragen

■ Vermeiden von dynamischem SQL mit Eingabewerten

- Zusammensetzen der SQL-Anweisung und der Eingabewerte als Zeichenfolge
- Ausführen der so aufbereiteten Zeichenfolge als SQL-Anweisung
- Zu finden in Access-Applikationen, Web-Applikationen, Einsatz von O/R-Mappern
- Kürzere Entwicklungszeit im Vergleich zu parametrisierten Abfragen

■ Parametrisierte Abfragen

- Empfohlene Sicherheitsmaßnahme gegen SQL Injection nach STIG, CIS, OWASP
- Trennt die eigentliche SQL-Anweisung von den eingegebenen Werten
- Verwendet Platzhalter für die eingegebenen Werte
- Ersetzt die Platzhalter durch die Werte erst bei der Ausführung der SQL-Anweisung
- Verhindert die Interpretation der eingegebenen Werte als SQL-Anweisung
- Realisierbar mit Gespeicherten Prozeduren

SQL Injection / Sicherheitsmaßnahmen / **Parametrisierte Abfragen**

- **Demo**

- Sicherheitsmaßnahme Parametrisierte Abfrage mit DAO

■ Parametrisierte Abfrage in Access mit VBA und DAO

- Bietet keinen Schutz
- Keine parametrisierte Abfrage in VBA mit DAO möglich
- Erfordert Pass Through-Abfrage für Zugriff auf SQL-Backend wie SQL Server

■ Parametrisierte Abfrage in Access mit Pass Through-Abfrage

- Keine direkte Parameterübergabe an Pass Through-Abfrage möglich
 - Erfordert Anpassen des Quellcodes der Pass Through-Abfrage
- Erweitern des Quellcodes mit „Parameter“ nur per dynamischem SQL möglich
- Erlaubt das Ergänzen des Parameterwerts mit weiteren SQL-Anweisungen
 - Beispiel Kennwort: *mir egal' DROP TABLE Stellen SELECT 'A*
- Clientseitige Eingabeprüfung als mögliche Sicherheitsmaßnahme
- Keine Auswirkung bei serverseitiger Eingabeprüfung in der Gespeicherten Prozedur

SQL Injection / Sicherheitsmaßnahmen / Parametrisierte Abfragen mit ADO

- Demo

- Sicherheitsmaßnahme Parametrisierte Abfrage mit ADO und Parametern

SQL Injection / Sicherheitsmaßnahmen / Parametrisierte Abfragen mit ADO

■ Parametrisierte Abfrage in Access mit VBA und ADO

- Bietet geringen bis hohen Schutz
- Geringer Schutz bei Ausführung von EXECUTE und dynamischem SQL
 - Keine parametrisierte Abfrage
- Hoher Schutz bei Verwendung einer tatsächlich parametrisierten Abfrage
 - Verwenden des Command-Objekts mitsamt Definition der Eingabeparameter

■ ADO

- ActiveX Data Objects
- Datenzugriff für externe Datenbanken
- Direkter Zugriff auf Objekte der externen Datenbank
 - Benötigt keine Access-Objekte wie eingebundene Tabellen und Pass Through-Abfragen
- Erfordert Microsoft OLE DB-Treiber for SQL Server ~ aktuell: MSOLEDBSQL19

SQL Injection / Sicherheitsmaßnahmen / Parametrisierte Abfragen mit dynamischen SQL

■ Vermeiden von dynamischem SQL mit Strukturänderung

- Zusammensetzen der SQL-Anweisung als Zeichenfolge
 - Einfügen von Objektnamen wie Tabellen- und Spaltennamen
 - Erweitern der WHERE-Klausel mit Filterkriterien per AND und OR
 - Anpassen der Sortierkriterien und Sortierreihenfolge
- Ausführen der so aufbereiteten Zeichenfolge als SQL-Anweisung
- Zu finden in Access-Applikationen, Desktop-Applikationen, Web-Applikationen
 - Anwendungsfälle wie komplexe Suchdialoge, Mandantenfähige Abfragen etc.

■ Sicherheitsmaßnahmen

- Clientseitige und serverseitige Eingabeprüfung
- Serverseitige Bereinigung der Parameterwerte mit QUOTENAME
- Verwenden von sp_executesql zum Erstellen einer parametrisierten Abfrage

SQL Injection / Sicherheitsmaßnahmen / Parametrisierte Abfragen mit dynamischen SQL

- Demo

- Sicherheitsmaßnahme Parametrisierte Abfrage per ADO mit dynamischem SQL

SQL Injection / Sicherheitsmaßnahmen / Parametrisierte Abfragen mit dynamischen SQL

■ Serverseitige Eingabeprüfung

- Eingabeprüfung auf Längen und Verwendung verbotener Begriffe
- Bestenfalls eine ergänzende Maßnahme

■ Serverseitige Bereinigung der Parameterwerte

- Bereinigen der Parameterwerte mit QUOTENAME und Trennzeichen
- Verwandelt bei Eingabewerten und Hochkommata die SQL Injection in einen Parameterwert
- Interpretiert bei Objekten mit Brackets die SQL Injection als Objektname

■ sp_executesql

- Ausführen einer dynamischen SQL-Anweisung als parametrisierte SQL-Anweisung
 - Trennt die SQL-Anweisung von den Parameterwerten
- Objektnamen und Strukturänderung nicht parametrisierbar

SQL Injection / Sicherheitsmaßnahmen / **Berechtigungskonzept**■ **Berechtigungskonzept**

- Effektivste Sicherheitsmaßnahme gegen unerlaubte Zugriffe
- Separation Of Duties = Trennen der Aufgaben von den Personen
- Least Privilege = Erteilen der zum Erfüllen der Aufgaben erforderlichen Zugriffsrechte

■ **Umsetzung**

- Analyse und Definition der Prozesse und der Aufgaben der Personen
- Trennen der definierten Prozesse von den Personen
- Zusammenfassen der Personen in AD-Gruppen
 - Erstellen von Anmeldungen in der SQL Server-Instanz für die AD-Gruppen
- Zusammenfassen der Prozesse in Datenbankrollen
 - Erteilen der Zugriffsrechte für die Datenbankrollen an den Objekten der Datenbank
 - Zuordnen der Anmeldungen zur Datenbank und dort zu den Datenbankrollen

SQL Injection / Sicherheitsmaßnahmen / **Berechtigungskonzept**

- Demo

- Berechtigungskonzept

Fazit

■ SQL Injection

- Einschleusen von SQL-Anweisungen in die aktuell auszuführende SQL-Anweisung
- Erster veröffentlichter Angriff im Jahr 2000
- 2025 Platz 2 der 25 Most Dangerous Software Weaknesses der [CWE](#)

■ Angriffsziel

- Datensicherheit durch Datenextraktion und Manipulation bzw. Löschen von Daten
- Benannte Gefährdungslage im BSI IT-Grundschatz APP.4.3 Relationale Datenbanken

■ Sicherheitsmaßnahmen

- Speichern der Fehlerdetails in einer Tabelle und Ausgabe einfacher Fehlermeldungen
- Verwenden parametrisierter SQL-Anweisungen anstatt dynamischem SQL
- Erstellen eines Berechtigungskonzepts nach Separation of Duties und Least Privilege

Danke

**Heute ist nicht alle Tage.
Das Thema kommt wieder.
Keine Frage.**

Frei nach „Paulchen Panther“

Danke

Noch Fragen?

✉ *info@berndjungbluth.de*

Vielen Dank für die Aufmerksamkeit.